Hotel Project Plan

OVERVIEW
The project topic, as defined by the Group Project Administrators is as follows:

"Hotel: Ring around the World

Design a web application that uses the Google Maps API to show the distribution of Computer Lab Ring members around the world. Include a secure protocol such that (with mutual consent) two members can exchange contact details when they live within a specified travelling distance, or are visiting the same location. All personal data must otherwise be securely protected, under the control of a database administrator."

The Hotel team has six members. The technical responsibilities of the team have been divided up as follows:

- Kristian Glass – Low-level PHP
- Mattias Linnap – PHP
- Andrew Nelson – JavaScript, Google Maps API
- Andy Patterson – Database/SQL, Design (CSS/XHTML)
- Peter Shepherd – PHP
- Andrew Sweet – Project Manager, PHP

PROCESS – *Spiral Model (SM) and Feature Driven Development (FDD)*
The spiral model is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model. It provides the potential for rapid development of increasingly more complete versions of the software.

The following framework activities will be used in each spiral:

1. Communication - Start
    a. Meetings of all group members, chaired by project manager
    b. Discussion will be held covering:
        i. Overall progress (compared against the acceptance criteria for the project)
        ii. Revisions to the specification and direction for the next iteration in terms of feature sets that would be desirable to implement
        iii. Personnel distribution - Each group member will not necessarily have a particular module that he works on and is his sole responsibility. Group members will be assigned to different areas of the project based on time constraints and their skill sets, so as to best implement the acceptance criteria.

2. Planning, Modelling and Construction as defined by FDD (feeding in work products from previous iterations)*
    a. Develop an overall model (more shape than content)
    b. Build a features list (A list of features grouped into sets and subject areas, based on the initial specification, with bias toward features included in the acceptance criteria - 'mandatory features')
    c. Plan by feature (A development plan, again with emphasis on implementing mandatory features)
    d. Design, code, test and document** (sub iteration taking up most of the overall iteration)
        i. Design by Feature (a design package – sequences)
        ii. Build (and test) by feature (completed client-value/high priority function)
3. Deployment
    a. Delivery of viable code, testing results and documentation (and integration testing, if necessary)
    b. Submission of progress report (encompassing '3.a' and written notes from '3.b')
4. Communication – End
    a. Meetings of all group members, chaired by project manager
    b. Progress reports will be given
    c. Discussion will be held on positive and negative points of the previous iteration
    d. Suggestions for improvements to process will be considered and fed back into the process

** Developers will be expected to capture the requirements of a feature in unit tests before any actual feature code is written ('test-driven development'). Both unit tests and feature-code will be expected to be fully documented.*

*** 'Document' is taken to mean the production of code documentation (e.g, via phpDoc) and user documentation (e.g. how the user uses a particular feature).*

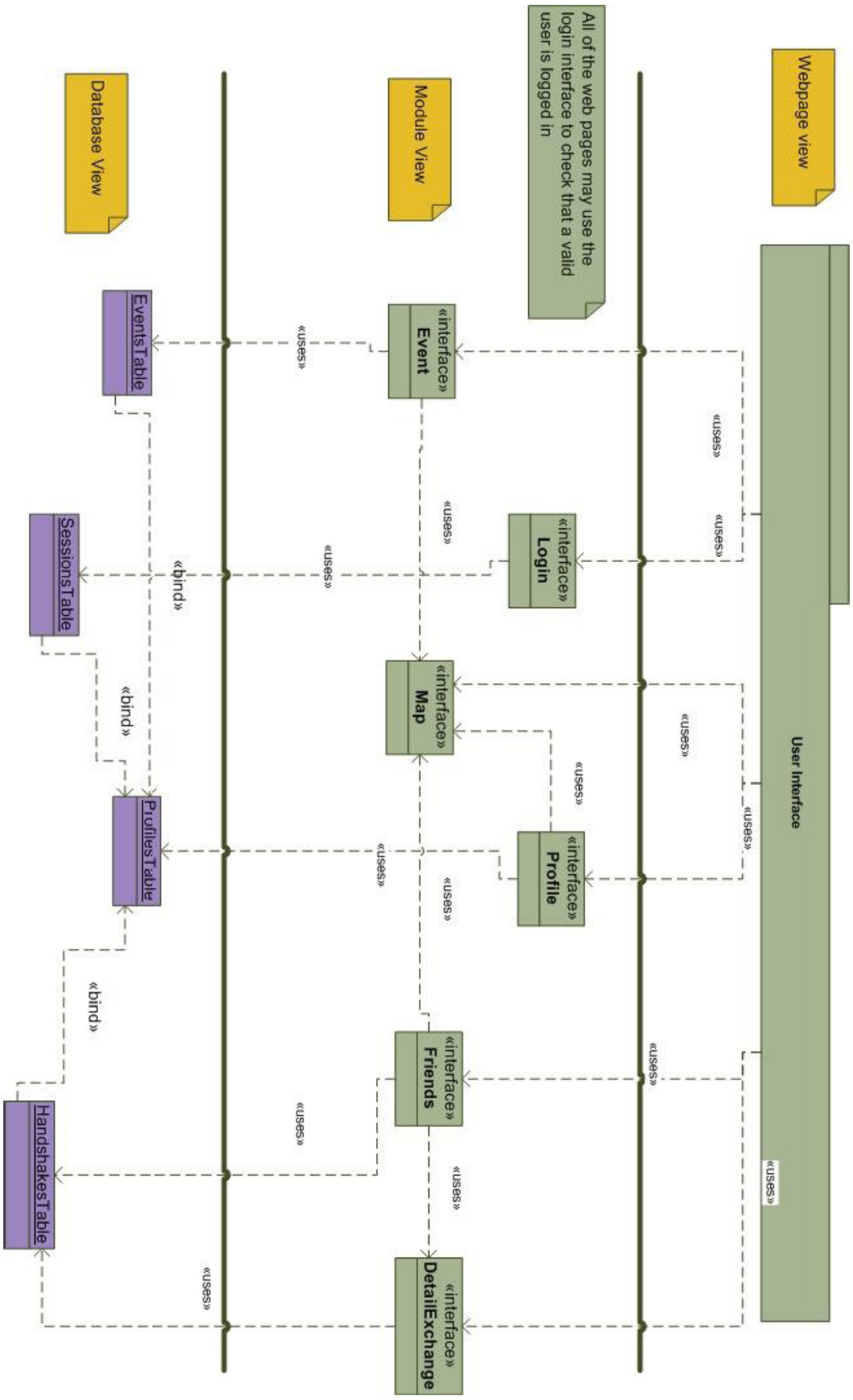
ARCHITECTURAL DESIGN

It is proposed that the system should be split into the following modules:

- Login
- Detail exchange
- Profile
- Event
- Maps
- Friends
- Layout

The web pages that users view will interact with one (or more) of these modules. In turn, the modules will interact with each other, and also with the

database. This design leads itself to a three-layered view of how the system will be constructed:



The public APIs of each of the modules (along with a suggestion for the generic 'Filter' class) are attached in the '.pdf' "Public APIs".

TIMETABLE

There will be three overarching iterations of the aforementioned spiral model, each ending with a named deliverable of the group project and a client meeting. The second and third iterations will each consist of three sub iterations, the first a one day iteration and the others five day iteration. Therefore, the timetable below simply details milestone goals that occur throughout the development process. Detailed timetabling for each (sub)iteration occur during the communication and planning phases of that iteration.

First Delivery – 18/01/07 to 29/01/07

The first delivery iteration started with the announcement of the group project allocations on Thursday 18th January and ends at 12pm on Monday 29th January.

Milestone goals:

- The development of a product specification detailing the:
  - major facilities to be provided - *Group*
  - major system components - *Group*
  - acceptance criteria for the finished product - *Group*
- An initial design model for the system will be developed – *Group*
- Work areas being divided between team members - *Group*
- Decisions regarding technologies, standards and the process(es) to be utilised in the project - *Group*

Deliverables:

- 'Functional Specification'
- 'Project Plan'

N.B. The details of both are listed in the group project handout at http://www.cl.cam.ac.uk/Teaching/GroupProjects/gpb0607/project_briefing_0607.pdf

Second Delivery – 29/01/07 to 12/02/07

The second delivery iteration will consist of three sub-iterations, as defined above ('a', 'b' and 'c' respectively).

Milestone goals:

- a) Set up PHP hosting and Subversion code repository and ensure that group members are able to access and use these resources – *Kristian, Andrew S*
- a) Develop unit tests for and implement abstract interface of each module's public API – *Module designers*

- a) Implement DB from first delivery design – *Andy P*
- Implement mandatory features of the following modules:
    - Login – *Mattias*
    - Maps – *Andrew N, Kristian*
    - Layout – *Andy P*
    - Events – *Pete*
- Determine viability of generic filtering subsystem and begin implementation if required – *Andrew S*

Deliverables:
- Group progress report – *Andrew S*
- Individual submissions – taking the form of unit test results and module documentation – *All*

Third Delivery - 12/02/07 to 26/02/07
The third delivery iteration will consist of three sub-iterations, as defined above ('a', 'b' and 'c' respectively).

Milestone goals:
- Implement mandatory features of all remaining modules:
    - Friends
    - Detail Exchange
    - Profile
- Implement preferable features as group sees fit
- Complete integration testing
- Complete final deployment

Deliverables:
- Group report – *Andrew S*
- Personal report – *All*
- All code, documentation, logs etc to be placed in ${CLTEACH}/grpproj by noon on 27/02/07 – *All*

IMPLEMENTATION TECHNOLOGIES

In order to achieve the maximum flexibility possible for the system, the implementation technologies were chosen for their multi-platform support both for installation/hosting and for execution/access.

PHP 5.2 – Hypertext Preprocessor
All server-side logic for the system will be written in PHP. PHP is an HTML-embedded scripting language. Much of its syntax is borrowed from C, Java and Perl with a couple of unique PHP-specific features thrown in. The goal of the language is to allow web developers to write dynamically generated pages quickly.

There exists a large library of open source packages for PHP known as PEAR - PHP Extension and Application Repository. From this library come two modules which will help with code documentation and testing.

Coding Standards - http://pear.php.net/manual/en/standards.php
The PEAR Coding Standards apply to code that is part of the official PEAR distribution. Coding standards often aim to keep code consistent to be easily readable and maintainable by developers. These standards will be utilised by Hotel for the project and all PHP code will be expected to conform to them.

Code Documentation – phpDocumentor – http://pear.php.net/package/PhpDocumentor
The phpDocumentor tool is a standalone automatic-documenter similar to JavaDoc and written in PHP. It differs from PHPDoc in that it is MUCH faster, parses a much
wider range of PHP files, and comes with many customizations including 11 HTML
templates, windows help file CHM output, PDF output, and XML DocBook peardoc2
output for use with documenting PEAR. In addition, it can perform PHPXref source
code highlighting and linking. It will be used to document any and all PHP code as per the recommendations in the phpDocumentor documentation.

Unit Testing – phpUnit - http://www.phpunit.de/
'phpUnit' is a member of the xUnit family of testing frameworks and provides both a framework that makes the writing of tests easy as well as the functionality to easily run the tests and analyse their results.

Client-Side JavaScript
JavaScript is the name of the Mozilla Foundation's implementation of the ECMAScript standard, a scripting language based on the concept of prototype-based programming. Client-side JavaScript (CSJS) is JavaScript that runs on client-side, i.e. the web browser, hence is for client-side scripting. While JavaScript was originally created to run on client-side, this term was coined because the language is no longer limited to just client-side, e.g. server-side JavaScript (SSJS) is also available. All client-side scripting in the Hotel project will be written in CSJS.

Unit testing will be attempted via *JsUnit - http://www.jsunit.net/* - and documentation via *JsDoc - http://jsdoc.sourceforge.net/*. However, due to the generally more variable nature of JavaScript, the developers will be greater leeway in how they implement testing and documentation, but the requirement to test and document will remain.

Database – MySQL 5.0
The MySQL database has become the world's most popular open source database because of its consistent fast performance, high reliability and ease of use. It's used in more than 10 million installations ranging from large

corporations to specialized embedded applications on every continent in the world (even Antarctica).

MySQL provides an API for access from PHP – mysqli (MySQL, Improved). This extension provides an advanced, object-oriented programming interface.